UNIVERSITY OF CALIFORNIA
RIVERSIDE

Experimental Implementation of Distributed Average Tracking for Heterogeneous Physical Agents Using Neighbors' Positions

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Electrical Engineering

by

Qianjun Liu

March 2019

Thesis Committee:
Dr. Wei Ren, Chairperson
Dr. Jianlin Liu
Dr. Sheldon Tan

The Thesis of Qianjun Liu is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

I would like to express my greatest gratitude to Professor Wei Ren of Electrical and Computer Engineering Department in University of California Riverside for his patience in answering my questions, giving me encouragement, pointing out my problems and helping me improve during my three semesters thesis work. Without his guidance, I am unable to complete this thesis.

I would also like to thank Prof. Sheldon Tan and Prof. Jianlin Liu for serving on my thesis committee. Their forward-looking comments have benefited me a lot in the thesis.

I would like to give my appreciation to Shan Sun for helping me on understanding the algorithms and the data collection during the experiments. I would like to thank Hanzhe Teng to guide me with the robot platform and help me with the experiment implantation. I want to say thank you to Pengxiang Zhu for his advice on time-varying system set up. I want to thank you Yifan Zhang also to help me carry the heavy robots from lab to the experiment ground in the cold nights.

Finally, I want to express my love to my parents for the love through my life.

ABSTRACT OF THE THESIS

Experimental Implementation of Distributed Average Tracking for Heterogeneous Physical Agents Using Neighbors' Positions

by

Qianjun Liu

Master of Science, Graduate Program in Electrical Engineering
University of California, Riverside, March 2019
Dr. Wei Ren, Chairperson

The focus of this thesis is on average tracking algorithm implementation for a group of heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics. In the algorithms, each agent is able to track the average of the time-varying reference inputs, where each agent has access to only its own position, its own input signals and the relative positions between itself and its neighbors. The algorithms are experimentally implemented on a multi-robot platform under an undirected communication topology. Simulation results and the experimental results based on the multi-robot platform are shown to validate the proposed algorithms. Finally, the error in simulation and experiment is analyzed based on experimental environment and robot characteristics.

Table of Contents

List of Figures **vii**

Chapter 1: Introduction **1**
1.1 Literature Review……2
1.2 Problem Statement……..……3
1.3 Outlines for Chapters……..……5

Chapter 2: Algorithms Introduction **7**
2.1 Notations and Preliminaries……7
2.2 Algorithms……...…….…..8

Chapter 3: Hardware and Software **10**
3.1 Hardware……...10
3.2 Software……12

Chapter 4: My work and Contribution **15**
4.1 The Multi-robot Experimental Platform Establishment and Improvement ….....15
4.2 The Connection Between Host and Client ……...……16
4.3 Theory Transformation into Practice ……...…17

Chapter 5: Architecture of Experimental Platform **21**
5.1 Multi-robot Experimental Platform based on Pioneer3AT……...21
5.2 Architecture of Host and Client Connection…….22
5.3 Experimental design on double-integrator and Euler-Lagrange dynamics……...23

Chapter 6: Simulation and Experimental Results **25**
6.1 Simulation ……25
6.2 Experimental Results……30

Chapter 7: Conclusion and Future Work **36**
7.1 Conclusions……...36
7.2 Future Work……...……...36

Bibliography **38**

List of Figures

1.1 Communication topology of agents……4

2.1 Sign function schematic diagram ……..……..…….7

3.1 One of the robots used in experiment……...10

3.2 The system hardware architecture……12

3.3 Simulation on MobileSim……13

4.1 The Multi-robot Experimental Platform in Coven Lab……16

4.2 A scene from the experiment……18

4.3 The hand position of the robot ……..……19

5.1 The network connection ……..……..……23

6.1 Positions of the agents and the average of the reference signals……...……...27

6.2 The average of agents' positions and the average of references ……..……28

6.3 The position of $x_i + \delta_i$ and the average of references…… ……...28

6.4 The error between the average of agents' positions and the average of references .29

6.5 The top view of the experiment ground ……..…….31

6.6 Positions of the robots and the average of the reference signals ……..…...…32

6.7 The average of robots' positions and the average of references …….....…….33

6.8 The position of $x_i + \delta_i$ and the average of references……...……...33

6.9 The error between the average of robots' positions and the average of references .34

Chapter 1

Introduction

Distributed average tracking has been a popular topic in recent years due to the significant research applied in the fields as autonomous driving, unmanned aerial vehicle, collaborative robot system. When the research keeps going, the situations that robots have more complex dynamics are needed to be faced.

This thesis presents the application on robot platform of the algorithm that for a group of heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics, each agent tracks the average of multiple time-varying reference inputs using only local information and local interaction of the agents. The algorithm is implemented in the ROS of Linux system based on Advanced Robot Interface for Application (ARIA). And the main framework and network connection used the tools in ARIA and was programed in C++. The formation and the algorithm application was programmed in python and applied to the multi-robot experimental platform built on Pioneer 3-AT robots. The simulation and the experiments are done for five heterogeneous physical agents under an undirected communication topology. At last, the simulation and the experiment results are provided to validate the algorithm.

1.1 Literature Review

Researchers have done excellent work on distributed average tracking for single-integrator systems [1-4], double-integrator systems[5], Euler-Lagrange systems [6] and nonlinear systems [7]. A proportional algorithm and a proportional-integral algorithm were proposed in [1] to achieve distributed average tracking for agents with single-integrator dynamics under constant or slowly-varying inputs. Then the proportional-integral algorithm was extended in [2] for distributed average tracking even in the presence of initial errors. In [3], the authors proposed a distributed nonsmooth control algorithm for a team of agent with single-integrators to track the average of multiple time-varying reference signals with bounded derivatives. Furthermore, the nonsmooth algorithm was extended into double-integrator dynamics [5] and Euler-Lagrange systems [6]. Recently, a linear distributed average tracking algorithm was proposed in [4] for single-integrator dynamics, which is experimentally implemented on a multi-robot platform. Considering more complex systems, the authors in [7] introduced a distributed average tracking algorithm for systems with heterogeneous unknown nonlinear systems. However, in practice, we might have to employ multiple robots with different abilities to accomplish task, so the situations that the group of robots have different dynamics are needed to be faced. Ref. 8 [8] addresses distributed average tracking for a group of heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics. Here, the goal if this thesis is to implement the algorithm in [8] on multi robot platform. In literature, ground robot platforms are most used to validate the proposed algorithms. In [9], considering distributed containment control, the authors showed both

simulation results and experimental results on a multi-robot platform to validate the theoretical results. Experimental results and the simulation on a formation control application are showed in [10] to validate one proposed proportion tracking algorithm. Distributed average tracking algorithm for single-integrator dynamics was implemented on multi-ground-robot platforms in [4].

The algorithms proposed in [8] about distributed average tracking for heterogeneous agents using neighbors' position are implemented and validated in this thesis. The algorithms will be simulated in MobileSim and tested on multi-robot experimental platform.

1.2 Problem Statement

This thesis introduced the situation of distributed average tracking for a group of heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics. The algorithms require each agent to have access to only its own position and the relative positions between itself and its neighbors. In this thesis, we consider five robots denotes by A, B, C, D and E where A, B are agents with double-integrator dynamics, C and E are agents with Euler-Lagrange dynamics and D is agent with single-integrator dynamics. The five agents are trying to track the average of multiple time-varying references.

Suppose there is a heterogeneous multi-agent system consisting of N physical agents. Where the index set has been defined as I $\{1, \ldots, N\}$. The agents are set as single-

integrator, double-integrator dynamics and Euler-Lagrange dynamics. With the consideration of the generality, the single-integrator agents are labeled as 1, …, $M-1$, where the dynamics is described by

$$\dot{x}_i = u_{i,} \qquad i = 1, \ldots, M-1 \tag{1}$$

The double-integrator agents are labeled as M, …, $N'-1$, with dynamics described by

$$\dot{x}_i = v_i, \;\; \dot{v}_i = u_i, \quad i = M, \ldots, N'-1 \tag{2}$$

Agents with Euler-Lagrange dynamics are labeled as N', …, N, with dynamics described by

$$M_i(x_i)\ddot{x}_i \; + \; C_i(x_i, \dot{x}_i)\dot{x}_i \; + \; g_i(x_i) \; = \; u_i, \;\; i \; = \; N', \ldots, N, \tag{3}$$

where $x_i(t) \in \mathbb{R}^{\mathfrak{p}}$ is ith agent's position, $v_i(t) \in \mathbb{R}^{\mathfrak{p}}$ is ith agent's velocity and $u_i(t) \in \mathbb{R}^{\mathfrak{p}}$ is ith agent's control input. $M_i(x_i)$ is the $\mathfrak{p} \times \mathfrak{p}$ symmetric inertia matrix, $C_i(x_i, \dot{x}_i)\dot{x}_i$ is the Coriolis and centrifugal force, and $g_i(x_i)$ is the vector of gravitational force. The Lagrange dynamics can be rewritten as in (1), i.e.,

$$M_i(x_i)\chi + \; C_i(x_i, \dot{x}_i)\psi \; + \; g_i(x_i) = \; Y_i(x_i, \dot{x}_i, \chi, \psi)\theta_i, \forall \chi, \psi \; \in \mathbb{R}^{\mathfrak{p}}, \tag{4}$$

where $Y_i \in \mathbb{R}^{\mathfrak{p} \times \mathfrak{p}_\theta}$ is the regression matrix and θ_i is the unknown but constant parameter vector.

The reference input $r_i(t) \in \mathbb{R}^{\mathfrak{p}}, \forall i \in I$ and its velocity $v_i^r(t) \in \mathbb{R}^{\mathfrak{p}}$ are bounded. Here the goal like in [1] is to design $u_i(t)$ for agent $i \in I$, to track the average of the reference inputs, i.e.,

$$\lim_{t \to \infty} \| \; x_i(t) - \frac{1}{N} \sum\nolimits_{j=1}^{N} r_j(t) \; \| = 0 \tag{5}$$

In this thesis, an average tracking problem for a team of heterogeneous agents is studied, where each agent uses local information to calculate the average of individual time-varying reference inputs, one per agent. The communication topology is shown in Fig. 1.1. They can only communicate with their neighbors as indicated in this topology.

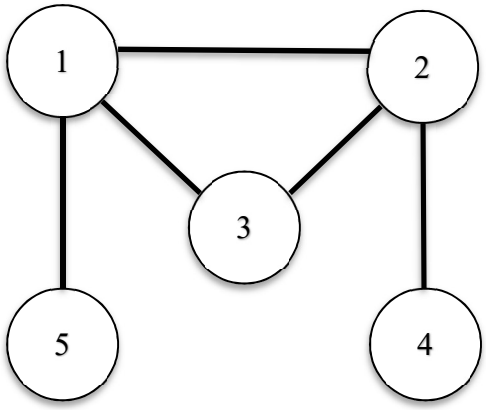

Figure 1.1: Communication topology of agents

1.3 Outlines for Chapters

Chapter 1 shows the related research on multi-agents tacking area and the problem statement of the algorithms to be implemented in this thesis.

Chapter 2 shows the mathematical expression of the algorithms derived in [8].

Chapter 3 is about the hardware and the software used in this thesis.

Chapter 4 is mainly about my work and contributions in the research.

Chapter 5 introduced the multi-robot experimental platform. It also includes the robot introduction used in experiments.

Chapter 6 provides simulation results, experimental results.

Chapter 7 covers the conclusion and future work.

Chapter 2

Algorithms Introduction

2.1 Notations and Preliminaries

Throughout the paper, $\mathbb{R}$ denotes the set of all real numbers. The transpose of matrix A and vector x are shown as A^T and x^T, and sgn(·) denotes the signum function defined componentwise like in Figure 2.1.

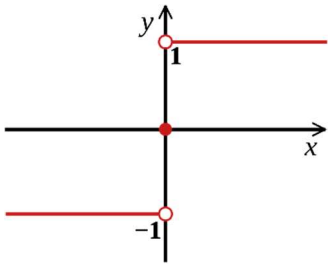

Figure 2.1 Sign function schematic diagram

In the framework, the topology is described by the undirected graph $\mathrm{G} \triangleq (V, E)$ is which is shown in Fig 1.1. $V \triangleq \{1, \ldots, N\}$ is the node set and $E \subseteq V \times V$ is the edge set.

2.2 Algorithms

To achieve goal (5), where each agent is required to have access to only its own position and the relative position between itself and its neighbors. In the following, the algorithms are described as in [1].

For agents with single-integrator dynamics, the control input u_i is designed as

$$u_i = -\beta_i \mathrm{sgn}\left[\sum_{j\in N_i}(x_i - x_j)\right] - (x_i - r_i) + v_i^r, \qquad i = 1,\dots,M-1 \tag{6}$$

For agents with double-integrator dynamics, the control input u_i is designed as

$$u_i = -\beta_i \mathrm{sgn}\left[\sum_{j\in N_i}(x_i - x_j)\right] - \sum_{j\in N_i}(x_i - x_j) - (x_i - r_i)$$
$$-2(v_i - v_i^r) + a_i^r, \qquad i = M,\dots,N'-1 \tag{7}$$

For agents with Euler-Lagrange dynamics, the control input u_i is designed as

$$u_i = Y_i(x_i, \dot{x}_i, q_i, \dot{q}_i)\hat{\theta}_i - \alpha s_i - \sum_{j\in N_i}(x_i - x_j),$$
$$\dot{p}_i = q_i$$
$$\dot{q}_i = -\beta_i \mathrm{sgn}\left[\sum_{j\in N_i}(x_i - x_j)\right] - (x_i - r_i) - 2(v_i - v_i^r) + a_i^r,$$
$$s_i = \dot{x}_i - q_i + x_i - p_i, \tag{8}$$
$$\dot{\hat{\theta}}_i = -Y_i(x_i, \dot{x}_i, q_i, \dot{q}_i)^T s_i, \qquad i = N'-1,\dots,N.$$

In (6), (7), (8), α and β_i are positive constant gains to be designed. sgn is the sign function. a_i^r is the acceleration of the reference. For (8), $\hat{\theta}_i$ is the estimate of the unknown but

constant parameters, s_i are defined as above. $Y_i(x_i, \dot{x}_i, v_i, v_i)$ is the regression matrix to rewrite the Lagrange dynamics as in (4). N_i is the set of ith agent's neighbors.

Additionally, as in [1], under the control law given by (6)-(8) for the system defined in (1)-(3), distributed average tracking goal (5) is achieved asymptotically, provided that assumptions that the topology graph G is connected and the reference input $r_i(t) \in \mathbb{R}^p, \forall i \in I$ and its velocity $v_i^r(t) \in \mathbb{R}^p$ are bounded The control gain β_i is chosen such that $min_{i\in I}\beta_i < \vec{r} + \vec{v_r}$ and $\alpha > 0$. For the equation here, the $\vec{r}$ is position vector of the reference, and the $\vec{v_r}$ is the velocity vector of reference.

Chapter 3

Hardware and Software

3.1 Hardware

The hardware listed below are what has been used in the thesis

1. Robots- The robots are Pioneer 3-AT which has an on-board computer. It is a four wheels drive robotic platform. The connection between the on-board computer and the motor has already been set up by the serial port connection. Besides, it is equipped with one battery, emergency stop switch, wheel encoders and a microcontroller with ARCOS firmware.

Figure 3.1 One of the robots used in experiment

2. On-board computer - The Pioneer 3AT robot is equipped with a Covalent Q87 mini-ITX single board computer. The Ubuntu 16.04 LTS system has installed on the on-board computer. The computer has the CPU as intel core 2 duo, P8400 and 2GB RAM. The disk size is 150GB.
3. WiFi-Router - The WiFi-Router is used to provide the center computer and on-board computer with the network.
4. Center Computer - A laptop with Ubuntu 16.04.3 LTS has been set as the central computer to send the center command to the five client robots which have been connected into the same network.
5. WiFi repeater – The WiFi repeater is used to extend the network to the experimental ground, in order to make sure the robot can assign the same local network as the host computer do.
6. Board car – The board car is used to carry the heavy robot from the lab room to the experimental ground.

As shown in figure 3.2, the hardware architecture is built on the multi-robot system with five robots, each robot has an on-board computer with Ubuntu 16.04. ARIA library, which is able to send command to the pioneer 3 AT robot, has been installed on the computer. So that the robot can drive the motor and wheels as the algorithm designed. At higher level, the laptop with Ubuntu 16.04 will connect to the same local network as the robots' on-board computers do through the WiFi Router.

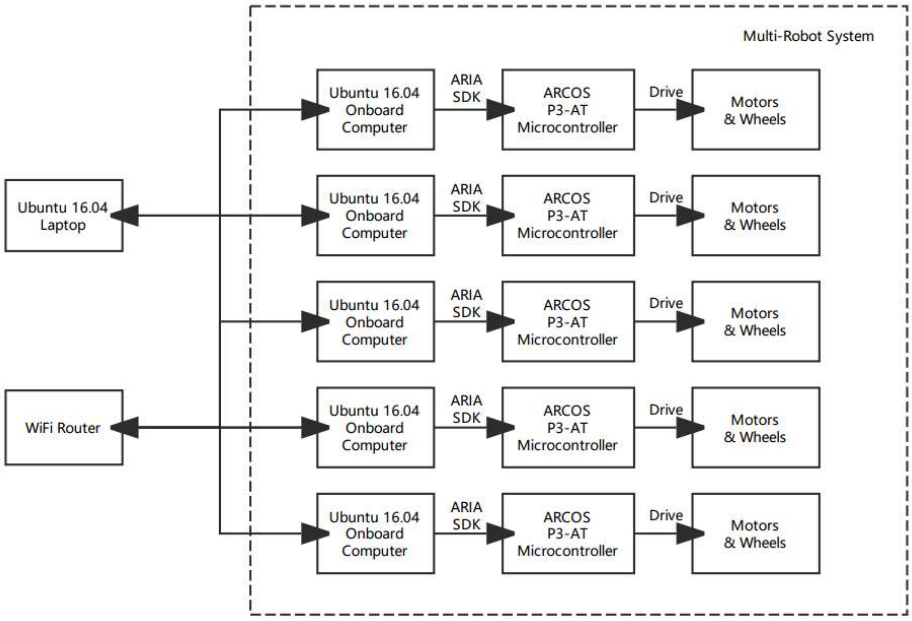

Figure 3.2 The system hardware architecture

3.2 Software

The software listed below are what has been used in the thesis

1. Operation system –Linux Ubuntu 16.04.3 LTS is installed on both the center computer and the on-board computers.
2. ARIA library – ARIA, programmed by C++, is Mobile Robots' Advanced Robot Interface for Application which has been offered by the pioneer 3. ARIA can dynamically control pioneer 3's velocity through simple low-level commands. ARIA also receives odometric position estimates sent by the robot platform.

3. MobileSim – It is a simulation software which has been built in the ARIA. It is used to debug the code, build the virtual environments, and test the algorithms.
4. ROS - The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a variety of robotic platforms.
5. TeamViewer – TeamViewer is the desktop remote control tool, the usage of it is easily login to the robot computer remotely.

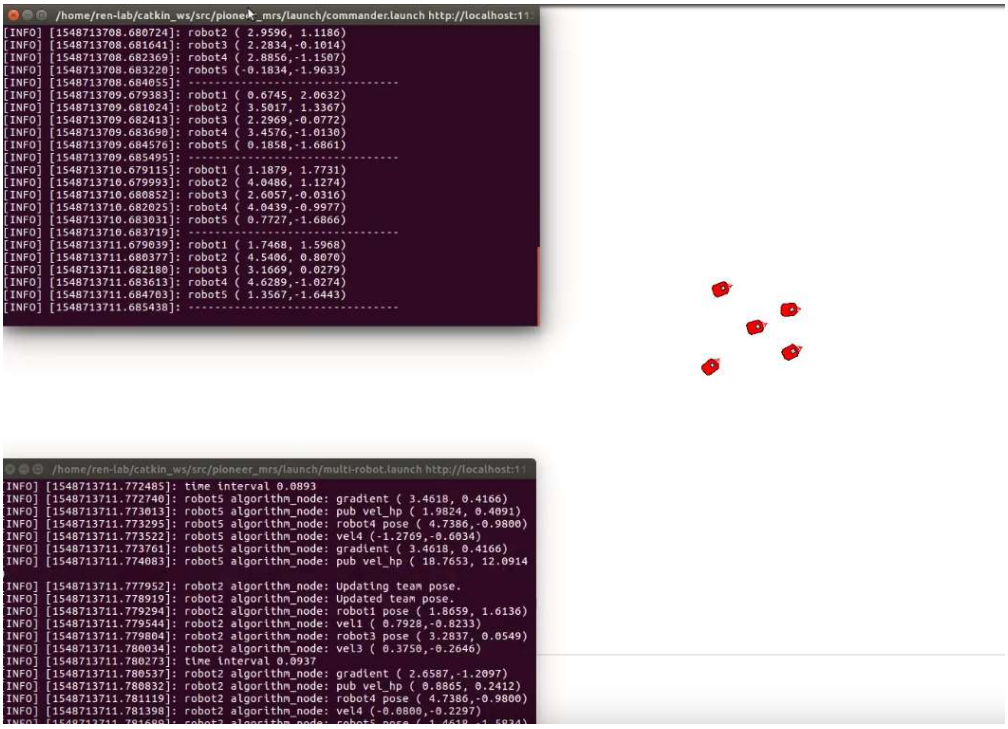

Figure 3.3 Simulation on MobileSim

The software architecture mainly follows the ROS Aria guide to set the library. During the setup of the ROS, first the source list file was created. After the installation, it needs to initialize the rosdep, the system cleaner, that the environment can be setup with the Aria workspace. And the source bash needs to be added into the login file which will save time to run that code every time. After the operating system and the ROS are ready on both the on-board computer and the host computer. The user should be put into the dialout group which will give the user the permission to get the access to serial port. Also, the SSH server configuration is modified that will make the login more convenient. Go through all the software using, the main work is to let the algorithm can be applied to the robot and capture the result and post out. For this direction the algorithm got through the ROS ARIA installed on the robot on-board computer to the pioneer 3 robot then drive itself to move. The result has been posted on the rostopic window.

Chapter 4

My work and Contribution

In this thesis, the robot experimental platform has been and modified, the connection between host and client has been established. The theoretical algorithms were transformed into practice. The simulation and the experimental results are given to validate the algorithms.

4.1 The Multi-robot Experimental Platform Establishment and Improvement

First, the Ubuntu system has been installed on the center computer and five on-board computers. The ARIA library which is the main C++ library used to control the robot is installed on the 5 on-board computers and the main computer. Based on the architecture introduced in 3.1, the system allows the 5 five robots run the algorithms at the same time and communicate with neighbors through the center computer. The position capture is done by the robot inbuilt odometer. The robot ROSARIA library gives the possibility to capture the angular velocity, the linear velocity and the pose as time goes on. Then time command in ROS is used to create a new time line for the system. The system subscribes the topic created by ROS about robot positions and velocities, which will be shown on the command

window. In order to avoid communication delay, the robots need to start working at the same time. That is the reason why a center computer is needed to guarantee each agent start the algorithm at the exact time. This part will be described in detail in chapter 4.2.

Figure 4.1: The Multi-robot Experimental Platform in Coven Lab

4.2 The Connection Between Host and Client

The connection between the host and the client should be setup before the running of the algorithm. The launch file is built to get the SSH access to each robot. At first, the easy code like the “SSH@” is used to log in the one robot to set the connection. Then robot will have access to the file and software in the target computer, but If the registration on different computer at different time, it just not fit the thesis requirement but make the robot into the different time line, which will let to the unfit reference and the

different velocity control. So the launch file is made to launch the five robots at the same time, for the launch file the laptop can just connect to the robots with the password, in other words, it means host have to login to make the connection before the launch step. To solve the problem here, host should be added into the robots' on-board computer free password list. The SSH key to add the target laptop into each robot onboard computer can realize the no -password login which will make the launch five robot at the same time realized. And the IP address was written as the designed ones on each computer include the host and the client. And all the network connection is under the same local net provided by the WiFi-Router. Finally, when the robot starts, agents will auto connect to the WiFi-Router, then the center computer can just easy launch at five robots. And the center computer is what only need to control during the algorithm running. The communication state program was built to set the undirected graph as the communication relations. It is been designed as a matrix whose dimensions are 5 for the easy change for the communication relation. During the launch, the launch information is designed as the number of the robot, the launch mode of the robot like doing the simulation on the MobileSim or launch the algorithm on the real robot, the position value of the robot and the communication port of the robot.

4.3 Theory Transformation into Practice

For validation of the theory, the theory needs to be transformed into practice. The reference is designed as $r_i = \left[\frac{0.5i}{3}t, 0.5icos\left(\frac{0.1}{3}t\right) - 0.5\right]^T$, i is the index number of

the robot from 1 to 5. Note that the references we designed have time-varying velocities and time-varying accelerations.

Figure 4.2: A scene from the experiment

The hand point based on feedback linearization will be introduced here. Because the robot has the control input as $(v, w)^T$, which has been considered as linear velocity and angular velocity. But the real handle position for the robot is not the center of the robot. A new hand position is set up for the robot instead of the center as the representation of the robot.

$$\begin{pmatrix} \dot{x}_h \\ \dot{y}_h \end{pmatrix} = \begin{bmatrix} \cos(\theta) & -L\sin(\theta) \\ \sin(\theta) & L\cos(\theta) \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{9}$$

$\dot{x}_h$ is the velocity of hand point on x axis. $\dot{y}_h$ is the velocity of hand point on y axis. θ is the constant parameters to represent angle as in the Figure 4.3. v is the constant parameters to represent linear velocity. ω is the constant parameters to represent angular velocity. L is the distance between the hand point with the center position informed in Figure 4.3.

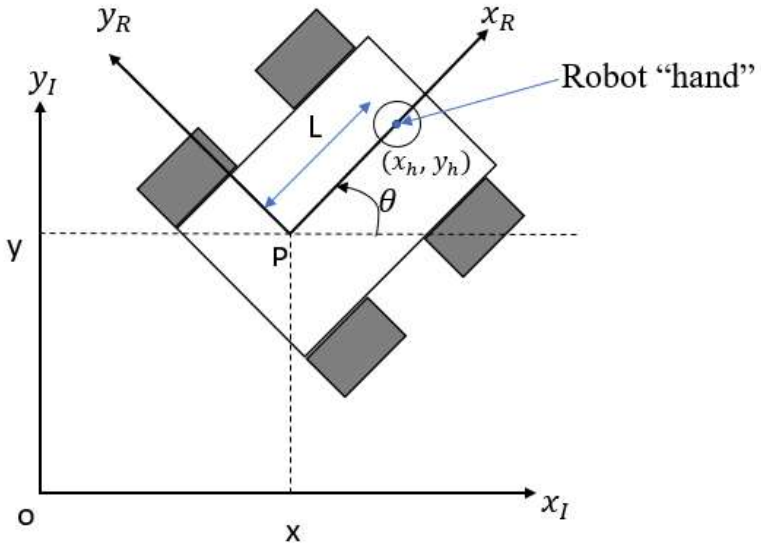

Figure 4.3 The hand point of robot

Based on the feedback linearization, a new hand position is calculated for the robot instead of the center as the representation of the robot. It can be assumed that the (x_i, y_i) as the inertial frame, (x_R, y_R) as the robot frame. The robot position be like $(x, y, \theta)^T$. For the differential drive robot, the kinematic equation.

$$\dot{x} = v\cos(\theta) \tag{10}$$

$$\dot{y} = v\sin(\theta) \tag{11}$$

$$\dot{\theta} = \omega \tag{12}$$

x is the position of robot on x coordinate. y is the position of robot assumed here on y coordinate. Form these equations as the transform matrix as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} \tag{13}$$

For the hand position, assume it as $(x_h, y_h)^T$, And the distance between the hand point with the center position is marked it as L in the Figure 4.3. After the homogeneous transformation, the coordinate representation should be like

$$P_h^I = \vec{OP}^I + R_R^I \vec{P}_h^R \tag{14}$$

$$= \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} L \\ 0 \end{bmatrix} \tag{15}$$

$$= \begin{bmatrix} x + L\cos(\theta) \\ y + L\sin(\theta) \end{bmatrix} \tag{16}$$

As the P_h^I is designed to be represent as $(x_h, y_h)^T$, the R_R^I is the transformation matrix from robot point to hand point. The derivation for the $(x_h, y_h)^T$ can should be like

$$\dot{x}_h = \dot{x} + L\big(-\sin(\theta)\big)\dot{\theta} \tag{17}$$

$$\dot{y}_h = \dot{y} + L\big(\cos(\theta)\big)\dot{\theta} \tag{18}$$

$$\begin{pmatrix} \dot{x}_h \\ \dot{y}_h \end{pmatrix} = \begin{bmatrix} \cos(\theta) & -L\sin(\theta) \\ \sin(\theta) & L\cos(\theta) \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{19}$$

$$\begin{bmatrix} \cos(\theta) & -L\sin(\theta) \\ \sin(\theta) & L\cos(\theta) \end{bmatrix}^{-1} u_i = \begin{bmatrix} v \\ w \end{bmatrix} \tag{20}$$

The hand position can be used as the control point to apply into the platform.

Chapter 5

Architecture of Experimental Platform

5.1 Multi-robot Experimental Platform based on Pioneer3AT

The Multi-robot experimental platform is built based on Pioneer3AT. Each robot has an embedded computer. The HDMI port to was used to connected to a monitor to edit files on the on-board computer. After that the SSH can be used to login the on-board computer from host computer. Mainly, like in Figure 3.2, the whole group of robots has connected to the WiFi-Router, like in Figure 4.1, there are five robots which used to form the experimental platform. When start a new algorithm, first is to start the robots make them power on. And agents will connect to the local network which is the same as the center computer has connected with automatically. Then the launch file can be run to make sure the connection has been built. Finally, algorithm can be started when make the inputs on the commander interface. During the algorithm running, the data such as position of each agents, the linear velocity and the angular velocity has been shown on the interface with the time line.

In conclusion, the multi-robot experimental platform is built on five Pioneer3 AT robots, each robot's on-board computer has installed with Ubuntu and set up with ARIA library. When the robots turn on the power, agents will automatically connect to the local network which is the same as the laptop connected. When apply the algorithm to the robot. It is only needed to send the command at the command window after launching the five robots.

5.2 Architecture of Host and Client Connection

The connection between host and client basic formed by SSH connections. The use of SSH is to log in each on-board computer from the center computer. The five robots IP address has been setup as designed (like 192.168.10.20X, X is the agent number from 1 to 5) And then the command will be sent to each agent to start the algorithms. Also, all the network has been provided from the local WiFi-Router. Like in Figure 5.1, the connections are active during the algorithm running. And once a connection is broken, it will be shown as the line lost in the figure. The figure is capture from the network function to check the connection. For example, the robot 1 check the communication state as the communication topology which will determine which the robot will have communication with itself. After getting the approve of communication with that robot, it will plug the information into algorithm node. It will exchange the action information based on the design algorithm. After calculating the hand point velocity out, it will pass the information to the pioneer server. The pioneer server got the position information from ROS and keep updating it to the server then to the algorithm to refresh the movement which has the real control speed of the robot. After the check and modify by ROS Aria which is the place to updating the

pose of the robot, the information of the robots will be sent back to the pioneer server. For the situation that needs to use the command window to input some movement to the robot, it will just send the command velocity to the server and the action topic to show the action of the robot. Which will make the robot move from the command which has been confirmed by pioneer server. And the pose change will be updating like above. The other four robots do the same steps likes this one. All of these formed the architecture of host and client connection.

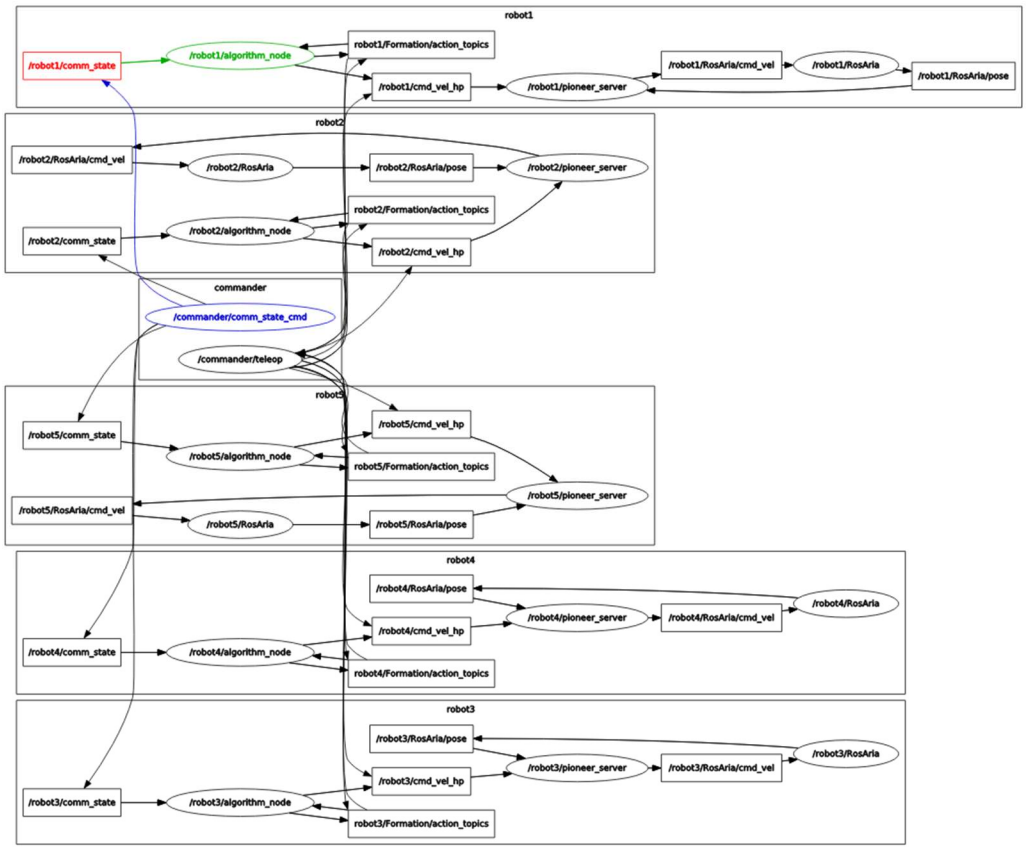

Figure 5.1 The network connection

5.3 Experimental design on double-integrator and Euler-Lagrange dynamics

The control input (7) and (8) for the system defined in (2) and (3) are both needs the control on the robots' acceleration. But the robot only has the control input as $(v, w)^T$. The solution here is that using the discrete integral control speed of the acceleration at very short intervals to replace the effect of the time-varying acceleration on the speed.

For the double-integrator dynamics which has been described as (2) can be rewritten as

$$\dot{x_i} = v_i,$$

$$v_i(k+1) = v_i(k) + u_i\hat{t}, \quad i = M, \ldots, N' - 1. \quad k = 0, \ldots T \tag{20}$$

For the Euler-Lagrange dynamics which has been described as (3) can be rewritten as

$$\dot{x_i} = v_i,$$

$$m_i\ddot{x_i}(k+1) = u_i - C_i v_i(k),$$

$$v_i(k+1) = v_i(k) + \ddot{x_i}(k+1)\hat{t}. \; i = M, \ldots, N' - 1. \quad k = 0, \ldots T \tag{21}$$

For (20) and (21), where the k means the index of the state of the system which has very short time interval. $\hat{t}$ is the short time interval which has been designed as $0.02s$.

Chapter 6

Simulation and Experimental Results

The simulation has been done on MobileSim which has the same setting and parameter design as the experiment. Because the robots in MobileSim has the exact robot's dynamics and characteristics as the real robot. For the real experiment, the experiment place should be big enough to contain all the robots and do not have large depressions and trenches. Actually, the ground is not horizontal. The flattest part of the ground was selected to decrease the influence. After the simulation which has been done on the MobileSim and the experiments has been done on the robots. the trajectories and relative data are shown. If the experiment is more ideal, the result will be closer to the result in simulation. The analysis was done on the comparison on trajectory and error with time-varying reference to validate the algorithm in chapter 2.

6.1 Simulation

In this section, simulation results are given to illustrate the effectiveness of the theoretical results. Assume that there are five agents (n = 5) in 2-D. The agent 1 and agent 2 is assigned to be the agents with double-integrator dynamics. The agent 4 is assigned to be the agent with single-integrator dynamics. The agent3 and agent 5 is assigned to be the agents with Euler-Lagrange dynamics. Suppose that the network topology is an undirected graph like Figure 1.1. The reference for agent i , $i = 1, \dots, 5$ is given by $r_i = \left[\frac{0.5}{3}it, 0.5icos\left(\frac{0.1}{3}t\right) - 0.5\right]^T$. The initial conditions of the agents are chosen as $x_1(0) =$

$[0.25,2.0]^T, x_2(0) = [0.25,1.0]^T, x_3(0) = [0.25,0.0]^T, x_4(0) = [0.25,-1.0]^T, x_5(0) = [0.25,-2.0]^T$, the control gain are chosen as $\alpha = 5$ and $\beta_1 = \beta_2 = 5$ which is assigned to the double-integrator dynamics, $\beta_3 = \beta_5 = 25$ which is assigned to the Euler-Lagrange dynamics $\beta_1 = 15$ which is assigned to the single-integrator dynamics, and $M_i = 12, C_i = 0.506$ for $i = 3,5$ was chosen as the official specification. Then an offset vector is introduced by replacing x_i with $x_i - \delta_i$ which will avoid agent crashed together. Here, $\delta_1 = [-1,1]^T, \delta_2 = [1,1]^T, \delta_3 = [0,0]^T, \delta_4 = [1,-1]^T, \delta_5 = [-1,-1]^T$, Figure 6.1 shows the positions of the agents and the average of the reference signals introduced and Figure 6.2 shows that the position of the center of agents and the average of reference. Clearly, all agents' positions track the average of the reference signals. Figure 6.4 shows the error between the average of robots and the average of reference. It is clearly to see that the error is bounded.

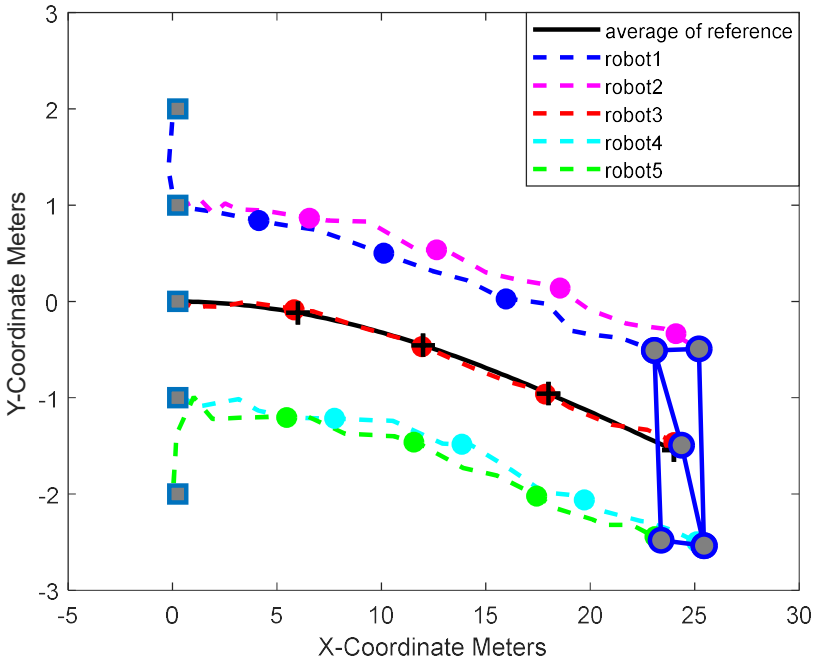

Figure 6.1 Positions of the agents and the average of the reference signals

Here, the positions are not initialized correctly, the agents' positions at every 12s are represented by dots on the dashed lines and the average of reference signals is plotted by a solid line. The squares denote the initial positions of the agents and circles represent their final positions. Note that the agents eventually form a square formation with its center tracking the average of the reference signals.

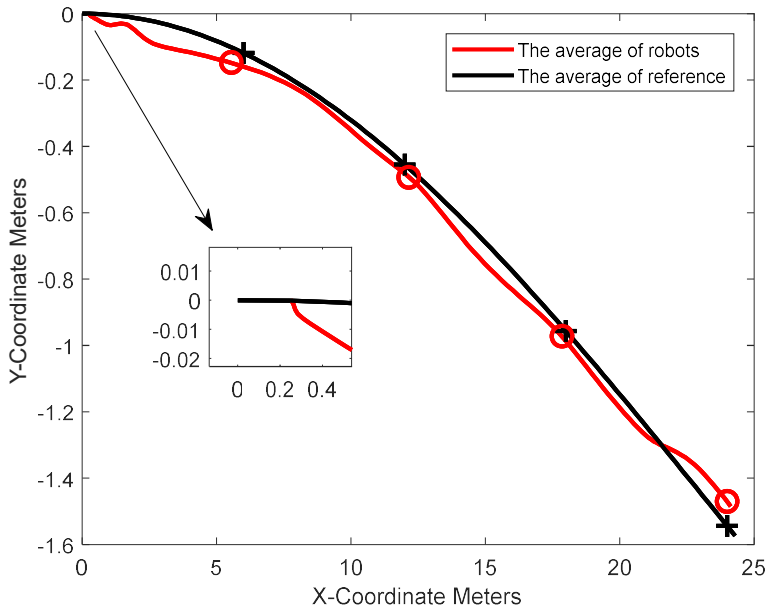

Figure 6.2 The average of agents' position and the average of references

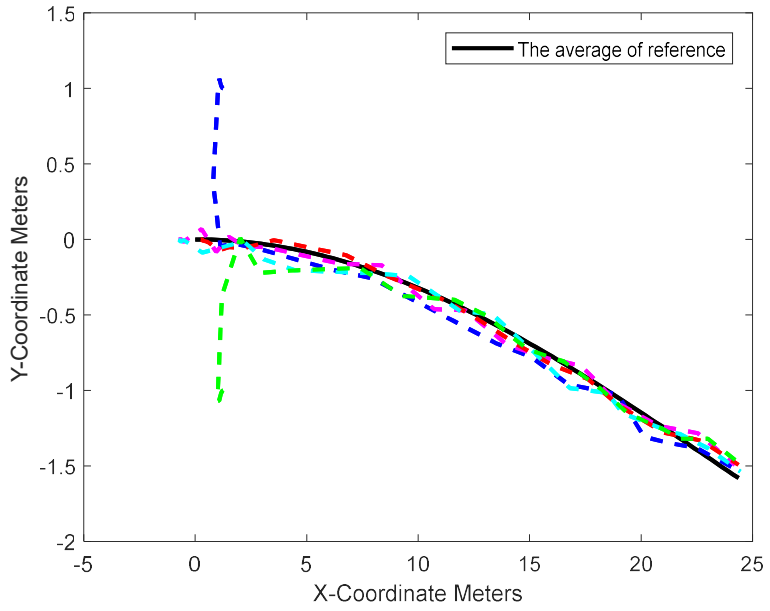

Figure 6.3 The position of $x_i + \delta_i$ and the average of references

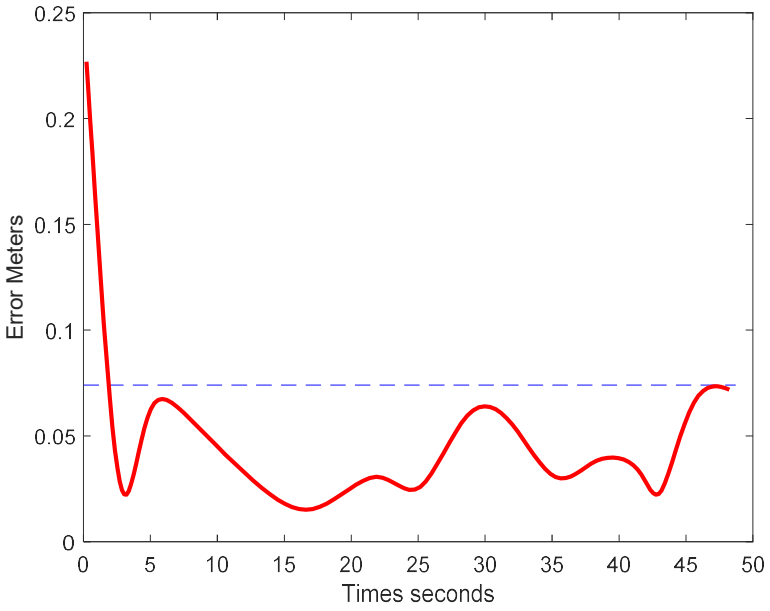

Figure 6.4 The error between the average of agents' positions and the average of reference

Noted that the initial point of the average of agent does not equal to the average of initial references, the average of agents' positions at every 12s are represented by circles on the lines and the average of reference's positions at every 12s are represented by plus on the lines It is clear to see that the average of the agents tracking the average the reference.

In Figure 6.3, the agents' positions with no offset $x_i + \delta_i$ and the average of the reference has shown as the dashed lines and the solid line. In Figure 6.4, The error is upper bounded by 0.084 as the blue line indicated. The cause of the error in simulation has the factor that, the latency in the ROS, the restriction of robot dynamics and the limitation of the control gain. For the latency in the ROS, when there are multiple process is running in the system at the same time, the ROS did not have the ability to control the system has the exact same feedback time, it sometimes make the same robot's information processed twice. Due to the Pioneer3-AT is the robots with four wheels, it is not the perfect model when set up the hand point as a differential drive robot. And because of the large control gain will make the robot shake frequently, we cannot increase the control gain for a better performance of the consensus of the robot's movement.

After the test and simulation on distributed average tracking for a group of heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics, the result shows that the algorithm is good to achieve the goal to let the average of the agent track the reference. After the analysis on the error, it is clearly that the error is bounded eventually.

6.2 Experimental Results

In this section, the introduced algorithms in chapter 2 are experimentally implemented and validated on a multirobot platform at outside experiment ground, the long wire is used to extend the WiFi-Router to the ground outside the Lab. Figure 6.5 shows the top view of the experiment ground. In the experiment, a $20 \times 10\, m^2$ area is used to implement the experiments. There are five robots (n = 5) in 2-D. The robot 1 and robot 2 is assigned to be the robots with double-integrator dynamics. The robot 4 is assigned to be the robot with single-integrator dynamics. The robot3 and robot 5 is assigned to be the robots with Euler-Lagrange dynamics. The network topology is an undirected graph like Figure 1.1. The reference for robot i, $i = 1, \dots, 5$ is given by $r_i = \left[\frac{0.5}{3} it, 0.5icos\left(\frac{0.1}{3} t\right) - 0.5\right]^T$. The initial conditions of the robots are chosen as $x_1(0) = [0.248, 2.0]^T, x_2(0) = [0.254, 0.990]^T, x_3(0) = [0.261, -0.001]^T, x_4(0) = [0.255, -1.0]^T, x_5(0) = [0.245, -2.001]^T$, the control gain are chosen as $\alpha = 5$ and $\beta_1 = \beta_2 = 5$ which is assigned to the double-integrator dynamics, $\beta_3 = \beta_5 = 25$ which is assigned to the Euler-Lagrange dynamics $\beta_1 = 15$ which is assigned to the single-integrator dynamics , and $M_i = 12, C_i = 0.506$ was chosen as the official specification. Then an offset vector is introduced by replacing x_i with $x_i - \delta_i$ which will avoid robot crashed together. Here, $\delta_1 = [-1, 1]^T, \delta_2 = [1, 1]^T, \delta_3 = [0, 0]^T, \delta_4 = [1, -1]^T, \delta_5 = [-1, -1]^T$, Figure 6.6 shows the Positions of the robots and the average of the reference signals introduced and Figure 6.7 shows that the position of the average of robots and the average of reference. Clearly, all robots' positions track the average of the reference signals. Figure 6.9 shows

the error between the average of robots and the average of reference. It is clearly to see that the error is bounded.

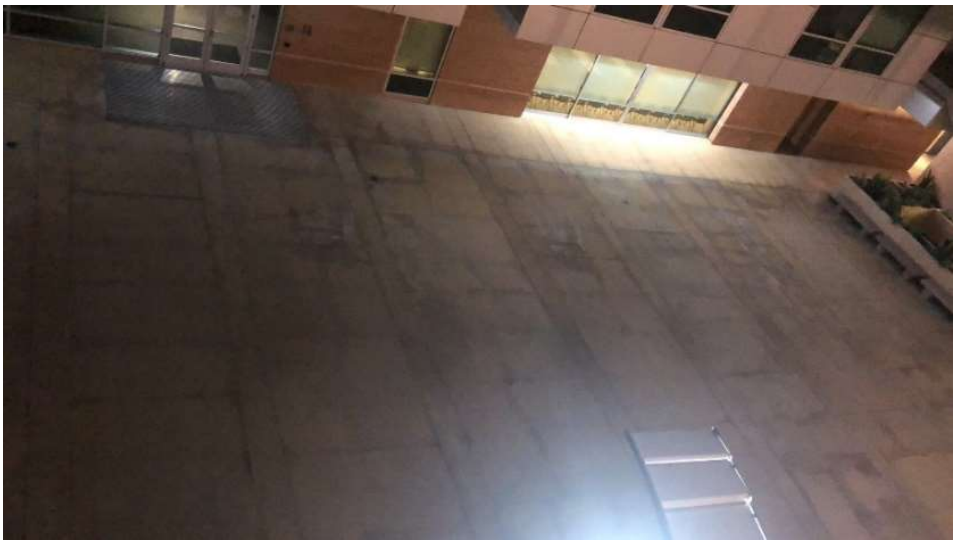

Figure 6.5 The top view of the experiment ground

In figure 6.6, the positions are not initialized correctly, the robots' positions at every 7.5 s are represented by dots on the dashed lines and the average of reference signals at every 7.5 s are represented by plus on the solid line. The squares denote the initial positions of the robots and circles represent their final positions. Note that the robots eventually form a square formation with its center tracking the average of the reference signals.

Noted that the initial point of the average of agent does not equal to the average of initial references, the robots' positions at every 7.5s are represented by circles on the lines and the average of reference's positions at every 7.5s are represented by plus on the lines It is clear to see that the average of the agents tracking the average the reference. In

figure 6.8, the robots' positions with no offset $x_i + \delta_i$ and the average of the reference has shown as the dashed lines and the solid line.

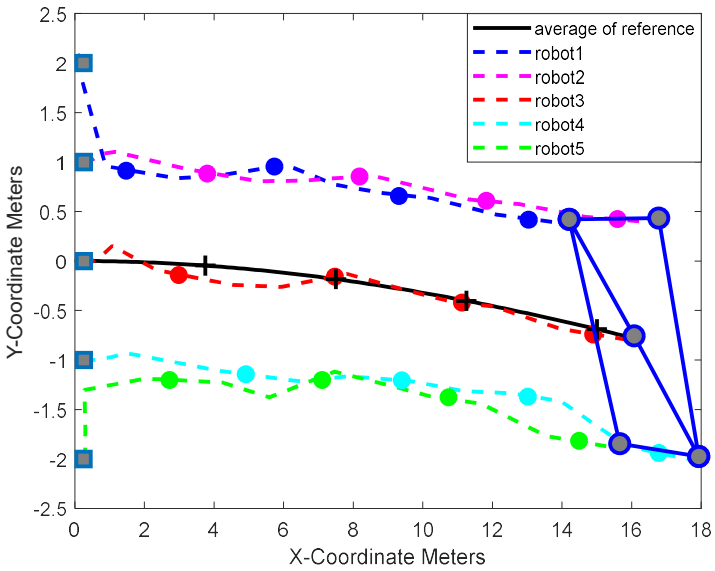

Figure 6.6 Positions of the robots and the average of the reference signals

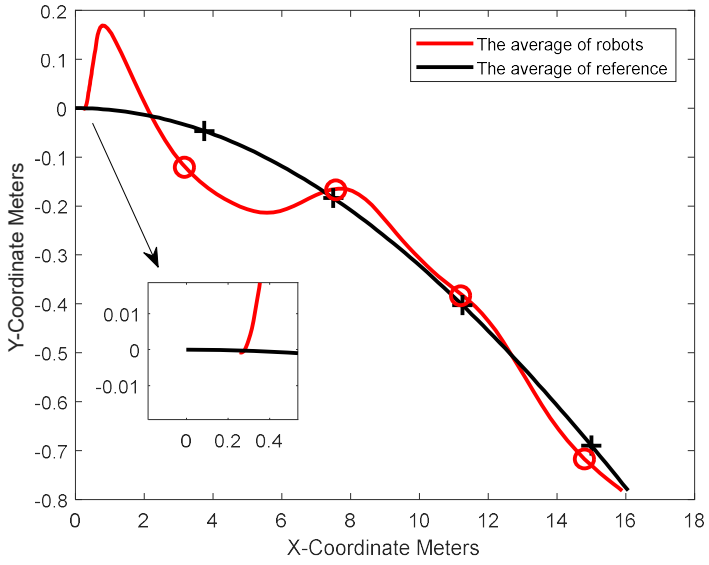

Figure 6.7 The average of robots' positions and the average of references

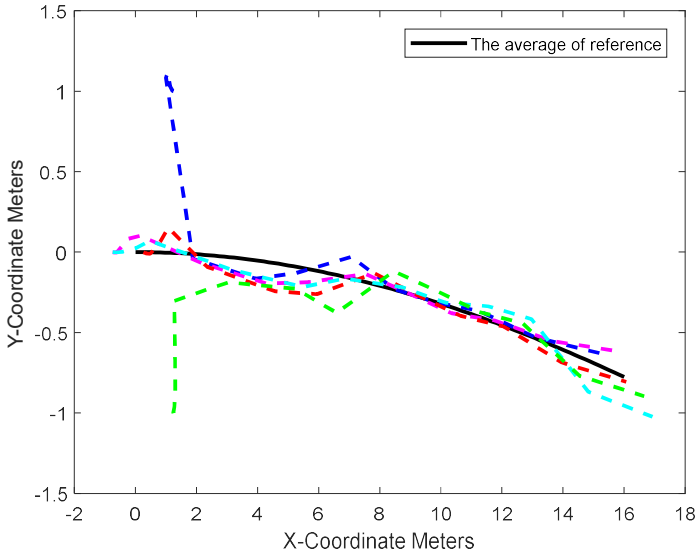

Figure 6.8 The position of $x_i + \delta_i$ and the average of references

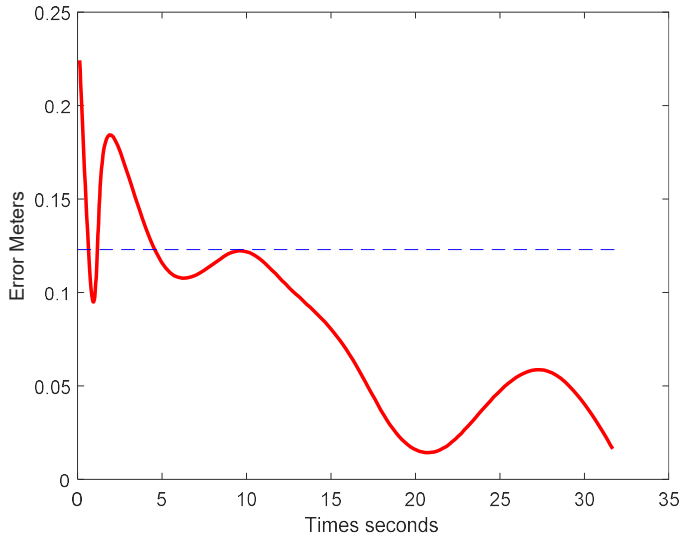

Figure 6.9 The error between the average of robots' positions

and the average of reference

In Figure 6.9, The error is upper bounded by 0.126 as the blue line indicated. For the error in the experiment, the factor cause the error in the simulation as the latency in the

ROS, the restriction of robot dynamics and the limitation of the control gain should also be considered. Also, there are more factor cause the error as below should be considered in the experiment. First, the experiment ground is not horizontal line, the angle between the X-Coordinate and Y-Coordinate is 10° larger than standard. Second, the ground friction is uneven, resulting in occasional slippage. Every time the robot turns, the tire and the ground will produce sliding friction, which makes error when the ground friction is uneven. The speed of the direction on y-axis and the influence of the slope will cause the coordinates of the odometer to be inaccurate. Different frictional forces in different areas will cause the parameters of the time-varying system can't be a fixed setting. At the same time, in the real experiment the communication delay caused by other usage of the robot WiFi router and the different process running in the host ROS system will cause the time mistake.

After the experiment on distributed average tracking for a group of heterogeneous robots consisting of single-integrator, double-integrator and Euler-Lagrange dynamics, the result shows that the algorithm is good to achieve the goal to let the average of the agent track the average of the reference input. After the analysis on the error, it is clearly that the error is bounded eventually.

Chapter 7

Conclusion and Future Work

7.1 Conclusions

This thesis studies the distributed average tracking for heterogeneous physical agents consisting of single-integrator, double-integrator and Euler-Lagrange dynamics using neighbors' positions. It shows how the algorithm has been implemented on the multi-robot experimental platform. It applied the algorithm theory into the real robots. It also describes the problem met in the experiment and the solutions. For the practical use of the robot. Based on the robot, the hand point is designed by feedback linearization. By setting the SSH log in, the connection has been realized between the host and the client. After the simulation and the experiments, the result and analysis about comparison on error is shown. The experiments are successful to reach the goal. The algorithm is validated in the robot platform.

7.2 Future Work

There are many researches on this area nowadays. There are three directions for the future work. First direction is focusing on the dynamics of heterogeneous agents tracking. The other dynamics can be added into the group of the robots. This will make the system fit more possible situations. When the groups of agents have more different

dynamics, the algorithms will change to more general. The second direction is the different types of the multi-robot platform, such as Crazy Flies, the different types of robot platform indicated the different use of the algorithm. Like work in the air instead of working on the ground. And like if the system processing under the water. It may have water robot platform. The third direction is the number of the robots. Crazy Flies can have more than 30 robots working together. Because the topology is simple and less connections, the experimental application on larger groups of robots may have good performance.

Bibliography

[1] R. A. Freeman, P. Yang and K. M. Lynch, “Stability and Convergence Properties of Dynamic Average Consensus Estimators,” in *Proceedings of the IEEE Conference on Decision and Control*. San Diego, 2006

[2] H. Bai, R. Freeman, and K. Lynch, “Robust dynamic average consensus of time-varying inputs,” in *Proceedings of the IEEE Conference on Decision and Control*. Atlanta, 2010

[3] F. Chen, Y. Cao and W. Ren, "Distributed Average Tracking of Multiple Time-Varying Reference Signals With Bounded Derivatives," in *IEEE Transactions on Automatic Control*, vol. 57, no. 12, pp. 3169-3174, Dec. 2012.

[4] M. Saim, S. Ghapani, W. Ren, K. Munawar and U. M Al-Saggaf, “ Distributed Average Tracking in Multi-Agent Coordination: Extensions and Experiments” in IEEE System Journals, vol. 12, no. 3, pp. 2428-2436, September 2018

[5] F. Chen, W. Ren, W. Lan and G. Chen, "Distributed Average Tracking for Reference Signals With Bounded Accelerations," in *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 863-869, March 2015.

[6] F. Chen, G. Feng, L. Liu and W. Ren, "Distributed Average Tracking of Networked Euler-Lagrange Systems," in *IEEE Transactions on Automatic Control*, vol. 60, no. 2, pp. 547-552, Feb. 2015.

[7] S. Ghapani, S. Rahili, and W. Ren, “Distributed average tracking for second-order agents with nonlinear dynamics,” in *American Control Conference*, Boston, MA, July 2016

[8] S. Rahili, W. Ren, “Heterogeneous Distributed Average Tracking Using Nonsmooth Algorithms,” *2017 American Control Conference*, May 2017, pp. 691-696.

[9] Y. Cao, D. Stuart, W. Ren, and Z. Meng, “Distributed Containment Control for Multiple Autonomous Vehicles with Double-integrator Dynamics: Algorithms and Experiments,” *IEEE Transactions on Control Systems Technology*, vol. 19, no. 4, July 2011, pp. 929-938.

[10] W. Ren, “Consensus Tracking under Directed Interaction Topologies: Algorithms and Experiments,” *IEEE Transactions on Control Systems Technology*, Vol. 18, No. 1, January 2010, pp. 230-237.

[11] S. H. Spong, Mark W. and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ: John Wiley and Sons, 2006